WEST Search History

Hide Items | Restore | Clear | Cancel

DATE: Monday, October 25, 2004

Hide?	Set Name	Query	Hit Count
		DB=USPT,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR	
☐	L38	l31 and l35	2
☐	L37	L35 and hash$4	0
☐	L36	L35 and (hash$4 near5 code)	0
☐	L35	L34 and (reset$4 with (control adj register))	36
☐	L34	execut$4 near3 bios near3 code	291
☐	L33	L32 same bios	31
☐	L32	L31.ab.	165
☐	L31	boot$4 adj sequence	776
☐	L30	boot$4 same (digest adj value)	2
☐	L29	boot$4 with (digest adj value)	1
☐	L28	L27 with hash$4	2
☐	L27	security with boot$4	716
☐	L26	bios near2 digest near2 value	0
☐	L25	l3.ab. and L18	4
☐	L24	l3.clm. and L18	12
☐	L23	l3 and L18	23
☐	L22	l6.ab. and L18	3
☐	L21	l6 and L18	47
☐	L20	l7 and L18	6
☐	L19	l10 and L18	8
☐	L18	l16 or L17	3559
☐	L17	713/2,100,200.ccls.	2824
☐	L16	710/16,104.ccls.	862
☐	L15	(fresh or clean) near3 restart$4 near3 boot$4	2
☐	L14	(nonboot$4 near2 device) or ("not" near2 boot$4)	10
☐	L13	(nonboot$4 near2 device)	10
☐	L12	(skip$4 or bypass$4) with (nonboot$4 near2 device)	0
☐	L11	(skip$4 or bypass$4) near3 nonboot$4 near2 device	0
☐	L10	((restart$4 or start$4 or begin$4 or initiat$4 or activat$4) near2 boot$4) same (skip$4 or bypass$4)	25

☐	L9	((restart$4 or start$4 or begin$4 or initiat$4 or activat$4) near5 boot$4) same (bypass$4 near5 device)	3
☐	L8	((restart$4 or start$4 or begin$4 or initiat$4 or activat$4) near5 boot$4) near5 (bypass$4 near5 device)	1
☐	L7	((restart$4 or start$4 or begin$4 or initiat$4 or activat$4) near5 boot$4) near5 ((based or depend$4 or accord$4 or respons$4) near5 device)	28
☐	L6	((restart$4 or start$4 or begin$4 or initiat$4 or activat$4) near5 boot$4) near5 (based or depend$4 or accord$4 or respons$4)	169
☐	L5	((fresh or clean) near5 (restart$4 or start$4 or begin$4 or initiat$4 or activat$4) near5 boot$4)	9
☐	L4	L3 same ((restart$4 or start$4 or begin$4 or initiat$4 or activat$4) near5 boot$4)	6
☐	L3	(determin$4 near5 (boot$4 adj device))	43
☐	L2	(boot$4 adj device) same (nonboot$4 adj device)	1
☐	L1	((initiat$4 or activat$4 or start$4) near3 boot$4)	2609

END OF SEARCH HISTORY

L14: Entry 1 of 10 File: USPT Jul 13, 2004

DOCUMENT-IDENTIFIER: US 6763454 B2
TITLE: System for allocating resources in a computer system

Detailed Description Text (46):
Unlike the system board devices and the boot-level devices connected to the integrated bus 15, which are respectively configured during steps 42 and 64, the remaining devices 20 connected to the integrated bus 15 are configured only after the boot operations for the computer 8 have been completed. However, the configuration operations for the nonboot-level devices on the integrated bus 15 are supported by the collection of device information that occurred prior to the BOOT process of step 68. In particular, this configuration process is supported by the current inventory of both the system board devices and the set of devices 20 connected to the integrated bus 15.

Detailed Description Text (47):
At step 70, the device drivers for this set of nonboot-level devices are identified in response to the device information collected during the preboot operation. The device drivers are typically identified by accessing corresponding device-related information that is stored in the computer database 16 or by accessing a predefined file of the operating system 10.

Detailed Description Text (48):
The device information collected during the preboot operation further supports the allocation and the assignment of the resources 14 required by the nonboot-level devices on the integrated bus 15. In step 72, the resource requirements and dependencies for each of the nonboot-level devices 20 on the integrated system bus 15 are compared to the available resources 14. This comparison permits a determination of whether a potential resource conflict exists. In an iterative fashion, potential resource conflicts are arbitrated and resolved prior to resource allocation. In step 74, the resources 14 are allocated to the nonboot-level devices 20 based upon the arbitration results of the step 72 and those devices are configured in step 76. In view of the allocated resources, the identified device drivers are loaded in step 78 and the devices are enabled for operation with the computer 8. This arbitration process is described in more detail below with respect to FIGS. 7-10.

Detailed Description Text (49):
For the system board devices and the set of devices directly connected to the integrated bus 15, device information has now been collected and stored in volatile computer memory and, as required for newly installed devices, in the nonvolatile memory of the computer database 16. The device information for the devices 20 on the integrated bus 15 may identify one or more of the devices as another system bus 18 capable of supporting other connected devices 20. Device information has not yet been collected for the "children" devices of each system bus 18 connected to the integrated bus 15. Nevertheless, for the preferred embodiment, the tasks of identifying device drivers, arbitrating and allocating the resources 14, and loading the identified device drivers for the set of nonboot-level devices 20 on the integrated bus 15 enable the subsequent collection of device information from these children devices. Thus, at step 80, an inquiry is conducted to determine whether any of the devices 20 on the selected bus are operable as system busses. If

not, the "NO" branch is followed to step 82, and the automated configuration operation for the computer 8 is completed. In contrast, if another system bus is connected to the computer 8, then the "YES" branch is followed to step 84 to continue the data collection process.

Detailed Description Text (164):
Referring now to FIGS. 1, 4, and 7, the arbitration process begins at the START step 700 in response to an arbitrator 154 receiving a list of device configurations from the configuration manager 158. The configuration manager 158 can sort the list of device configurations based on the number of possible resource requirements for a particular type of resource. This sort, which is preferably completed for device configurations representing nonboot-level devices, is based on the complexity of the requirement for resource elements, as defined by each device configuration. The configuration manager 158 passes the list of device configurations to the arbitrator 154 representing the selected type of resource 14, such as the memory 22, interrupts 24, DMA channels 26, and I/O ports 28.

Previous Doc Next Doc Go to Doc#

L14: Entry 3 of 10 File: USPT Nov 13, 2001

DOCUMENT-IDENTIFIER: US 6317828 B1
TITLE: BIOS/utility setup display

CLAIMS:

4. The method according to claim 3, wherein comparing the preselected number of bootable devices against the number of BCVs in the list further comprises:

searching for a link;

determining a number of links that are found;

checking the number of links that are found against the preselected number of bootable devices;

if for a link the number of links that are found is greater than the preselected number of bootable devices allocated to the option ROM, storing an identifier of the link found for initialization as a nonbootable device for initialization; and

if the number of links that are found is less than the preselected number of bootable devices allocated to the option ROM, storing an identifier of the link in the list of BCVs.

15. The computer system of claim 14, wherein the system BIOS ROM code that compares the preselected number of bootable devices against the number of BCVs in the list further searches for a link, determines the number of links found, checks the number of links found against the preselected number of bootable devices, and if the number of links found is greater than the preselected number of bootable devices allocated to the option ROM, stores the link for initialization as a nonbootable device for initialization, and if the number of links found is less than the preselected number of bootable devices allocated to the option ROM, stores the link in the list of BCVs.

26. The computer program product of claim 24 wherein comparing the preselected number of bootable devices against the number of BCVs in the list further comprises:

searching for a link;

determining a number of links that are found;

checking the number of links that are found against the preselected number of bootable devices;

if for a link the number of links that are found is greater than the preselected number of bootable devices allocated to the option ROM, storing an identifier of the link found for initialization as a nonbootable device for initialization; and

if the number of links that are found is less than the preselected number of bootable devices allocated to the option ROM, storing an identifier of the link in

L7: Entry 1 of 28 File: USPT Jul 13, 2004

DOCUMENT-IDENTIFIER: US 6763454 B2
TITLE: System for allocating resources in a computer system

Detailed Description Text (41):
In step 54, an inquiry is conducted to identify the subset of the devices 20 that must be active upon completion of the boot process. For the devices 20 that do not require a default-type configuration during the power-up sequence, the "NO" branch is followed to the step 56 and those devices preferably remain inactive during the power-up sequences. In contrast, the "YES" branch is followed from the step 54 to the step 58 for the devices 20 that must be activated during the boot process. Based upon this inventory of the identified devices 20 requiring activation during the boot process, a boot-level device driver for each of those devices is obtained in step 58 to enable communications between the boot-level devices and the computer 8. These boot-level devices typically include the system-level devices 20 on the system board of the computer 8 and certain adapter boards connected to the integrated bus 15, such as a display controller or a mass memory storage device controller, i.e., a fixed disk controller.

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L7: Entry 6 of 28 File: USPT Jun 4, 2002

DOCUMENT-IDENTIFIER: US 6401183 B1
TITLE: System and method for operating system independent storage management

Detailed Description Text (3):
Prior art computer devices include pre-determined start-up or boot sequences which are based on the firmware level of the device. These sequences ultimately load the run-time operating system for that device, which, in turn, controls the device.

Previous Doc Next Doc Go to Doc#

First Hit Previous Doc Next Doc Go to Doc#

Generate Collection Print

L7: Entry 24 of 28 File: JPAB Oct 28, 1991

PUB-NO: JP403241448A
DOCUMENT-IDENTIFIER: JP 03241448 A
TITLE: IPL SYSTEM

PUBN-DATE: October 28, 1991

INVENTOR-INFORMATION:
NAME COUNTRY
TAKITA, MASATOSHI
KUWATA, SATORU
YOSHIOKA, ATSUSHI
OGASAWARA, YASUO

ASSIGNEE-INFORMATION:
NAME COUNTRY
FUJITSU LTD

APPL-NO: JP02039035
APPL-DATE: February 20, 1990

INT-CL (IPC): G06F 13/00; G06F 9/445

ABSTRACT:

PURPOSE: To share a processor by monitoring the occurrence of a response from a peripheral device after starting up a system, and generating a boot transfer request to a response generating device at the time of detecting the response.

CONSTITUTION: When an IPL request from the outside is inputted, the IPL processing part 10 of a central processing part 1 is started up, and a peripheral device monitoring means 101 is driven. The peripheral device monitoring means 101, when detecting a certain response or request from the peripheral device, identifies the peripheral devices as an IPL target device, and supplies a transfer instruction from a boot request means 102 to an identified peripheral device, and makes it execute the transfer of a boot program. In such a way, it is possible to respond to the IPL request from every peripheral device with a common control function.

Previous Doc Next Doc Go to Doc#

L2: Entry 1 of 1 File: USPT Nov 13, 2001

DOCUMENT-IDENTIFIER: US 6317828 B1
TITLE: BIOS/utility setup display

CLAIMS:

4. The method according to claim 3, wherein comparing the preselected number of bootable devices against the number of BCVs in the list further comprises:

searching for a link;

determining a number of links that are found;

checking the number of links that are found against the preselected number of bootable devices;

if for a link the number of links that are found is greater than the preselected number of bootable devices allocated to the option ROM, storing an identifier of the link found for initialization as a nonbootable device for initialization; and

if the number of links that are found is less than the preselected number of bootable devices allocated to the option ROM, storing an identifier of the link in the list of BCVs.

15. The computer system of claim 14, wherein the system BIOS ROM code that compares the preselected number of bootable devices against the number of BCVs in the list further searches for a link, determines the number of links found, checks the number of links found against the preselected number of bootable devices, and if the number of links found is greater than the preselected number of bootable devices allocated to the option ROM, stores the link for initialization as a nonbootable device for initialization, and if the number of links found is less than the preselected number of bootable devices allocated to the option ROM, stores the link in the list of BCVs.

26. The computer program product of claim 24 wherein comparing the preselected number of bootable devices against the number of BCVs in the list further comprises:

searching for a link;

determining a number of links that are found;

checking the number of links that are found against the preselected number of bootable devices;

if for a link the number of links that are found is greater than the preselected number of bootable devices allocated to the option ROM, storing an identifier of the link found for initialization as a nonbootable device for initialization; and

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L4: Entry 1 of 6 File: USPT Apr 13, 2004

DOCUMENT-IDENTIFIER: US 6721883 B1
TITLE: System and method for managing the boot order of a computer system

Brief Summary Text (6):
In prior computer systems, the boot order of the computer systems was determined by the computer system's scan order. The computer system scanned the PCI devices of the computer system, starting with the PCI bus and device having the lowest number. Thus, the computer system would first attempt to boot from device 0 on PCI bus 0. If the computer system determined that it could not boot from device 0 on PCI bus 0, the computer system would continue its scan for a bootable device at device 1 of PCI bus 0. Alternatively, the direction of the boot scan could be reversed so that the boot scan would begin at the highest numbered PCI bus and device. If the user wished to boot from a particular device, the user had to physically install the device in the appropriate device slot so that the device would be the first bootable device encountered during the boot scan. This method of specifying the boot device was disadvantageous in that it required the physical manipulation of the bootable devices as a means of specifying the boot order.

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L4: Entry 4 of 6 File: USPT Sep 18, 2001

DOCUMENT-IDENTIFIER: US 6292890 B1
TITLE: Computer system with dynamically configurable boot order

Abstract Text (1):
A computer system is provided with a dynamically reconfigurable boot order. In one embodiment, the computer comprises a network interface, a nonvolatile memory, and a CPU. The network interface may be coupled to a network to receive a "wake-up" data packet, that is, a data packet that includes a predetermined data pattern for which the network interface can be configured to scan. Upon detecting the wake-up data packet, the network interface can initiate a computer boot-up sequence. The CPU begins a boot-up sequence by retrieving a BIOS from the nonvolatile memory. One portion of the boot-up sequence specified by the BIOS includes determining and accessing a series of target boot devices to locate and retrieve an operating system. Preferably, the order of the series of target boot devices (i.e. the "boot order") is different from a default boot order if the network interface initiated the current boot-up sequence. Otherwise, the default boot order is used. The difference in the boot orders may be that the default boot order begins with local boot devices and the boot order resulting from a remote boot begins with network boot devices.

Brief Summary Text (15):
Accordingly, there is provided herein a computer system having a dynamically reconfigurable boot order. In one embodiment, the computer comprises a network interface, a nonvolatile memory, and a CPU. The network interface may be coupled to a network to receive a "wake-up" data packet, that is, a data packet that includes a predetermined data pattern for which the network interface can be configured to scan. Upon detecting the wake-up data packet, the network interface can initiate a computer boot-up sequence. The CPU begins a boot-up sequence by retrieving a BIOS from the nonvolatile memory. One portion of the boot-up sequence specified by the BIOS includes determining and accessing a series of target boot devices to locate and retrieve an operating system. Preferably, the order of the series of target boot devices (i.e. the "boot order") is different from a default boot order if the network interface initiated the current boot-up sequence. Otherwise, the default boot order is used. The difference in the boot orders may be that the default boot order begins with local boot devices and the boot order resulting from a remote boot begins with network boot devices.

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L5: Entry 1 of 9 File: USPT Mar 13, 2001

DOCUMENT-IDENTIFIER: US 6202153 B1
TITLE: Security switching device

Brief Summary Text (13):
Special software applications are able to detect attempts of such hostile intrusion to computer resources by unauthorized persons or data viruses. In this case, the computer has to be restarted (boot operation) from a "clean media" which is often called a rescue diskette, since the hard drives of the computer are suspected to be contaminated. This clean media often includes removable media such as a diskette or a CD-ROM.

Detailed Description Text (215):
According to a further aspect of the invention, the activation of the clean boot mode, as described in FIG. 16, can be done on a timely basis, at predetermined intervals.

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L5: Entry 2 of 9 File: USPT Jan 12, 1999

DOCUMENT-IDENTIFIER: US 5860140 A
TITLE: Circuit and method for learning attributes of computer memory

Brief Summary Text (22):
The present invention starts with a clean slate every time the computer is booted or rebooted. All memory is initially assumed to be noncacheable, nonwritable and writethrough and the mapping RAM reflects this fact. As the mapping circuit learns otherwise through successive memory accesses on behalf of the CPU, the mapping RAM will reflect this knowledge gained, and the system responds by operating more efficiently with respect to its memory management by tailoring its access as a function of the attributes. For instance, the CPU will not attempt to write to ROM and will allow caching of cacheable memory.

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L25: Entry 1 of 4 File: USPT Nov 4, 2003

DOCUMENT-IDENTIFIER: US 6643772 B1
TITLE: Universal boot code for a computer network

Abstract Text (1):
A method of and software for booting a network computer with universal boot code is disclosed. Initially, the type of a boot device is determined from among a set of possible boot devices. A command in a high level boot code segment of the boot code software is then translated to a command executable by the boot device based upon the determined device type. The converted command is then executed on the boot device to transfer data between the network computer and the boot device. The boot code is preferably compatible with a variety of boot devices including a hard disk boot device, an NFS server boot device, as well as a TFTP server boot device. In an embodiment in which the boot device is a TFTP boot device, a READ command from the high level boot code is translated to a TFTP read request. The data transferred by the TFTP read request may be stored in a file cache on the network computer. During a subsequent high level boot code READ command, the software interface may determine if the requested data is cached in the file cache, and if so, it may retrieve the data from the file cache. If the high level boot command is a SEEK command, and the boot device is a TFTP device, the converted command may include a TFTP read request. The software interface may determine the relative location of a file location indicated by the SEEK command and a current location of a file pointer and abort the current TFTP transfer if the file location indicated by the SEEK precedes the current location of the file pointer. The interface may then resend a TFTP read request to advance the file pointer to the file location indicated by the SEEK command. In this manner, the software interface and device specific segments can emulate a file type device when the boot device is a TFTP device.

Current US Original Classification (1):
713/2

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs
End of Result Set

Previous Doc Next Doc Go to Doc#

Generate Collection Print

L29: Entry 1 of 1 File: USPT Jun 6, 2000

DOCUMENT-IDENTIFIER: US 6073124 A
TITLE: Method and system for securely incorporating electronic information into an online purchasing application

Detailed Description Text (30):
Specifically, in step 801, the boot program reads the component list (the ".ssc" file) associated with the selected item of merchandise to determine what components to download from a specified content supplier server. In steps 802-808, the boot program executes a loop to process each remaining component in the component list that has not already been successfully downloaded. Specifically, in step 802, the boot program selects the next component from the component list that appears following the last successfully read component. In step 803, the boot program determines whether all of the remaining components of the list have been processed, and if so, returns, else continues in step 804. In step 804, the boot program determines whether the file indicated by the TRIGGER field is already present. If not, the boot program obtains the component indicated by the URI value from the content supplier server and stores the obtained component as indicated by the LOCAL value (see Table 2). In step 805, the boot program calculates a message digest (the value of a one-way hash function) for the downloaded component. In step 806, the determined message digest for the newly downloaded component is compared with a previously stored message digest in the component list (see the MSGDIG value in Table 2). In an exemplary embodiment, an MD5 algorithm is used to calculate a message digest. However, one skilled in the art will recognize that any message digest algorithm or any function capable of determining a predictable value for the downloaded component for comparison to an already stored value may be used. The MD4 and MD5 algorithms are described in Bruce Schneier, Applied Cryptography, John Wiley & Sons, Inc., 1994, which is hereby incorporated by reference. In step 807, if the calculated message digest is identical to the stored message digest, then the boot program continues in step 808, else continues back to the beginning of the loop in step 802, because a failure has occurred in downloading the component. In step 808, the boot program sets an indicator of the last successfully read component to indicate the component most recently loaded. In step 809, the boot program processes the component according to the tag (e.g., "Execute"), and continues back to step 802 to select the next component to download. Note that the tag associated with each component entry will automatically cause the secured content file (or the content player, depending on the situation) to begin executing.

Previous Doc Next Doc Go to Doc#